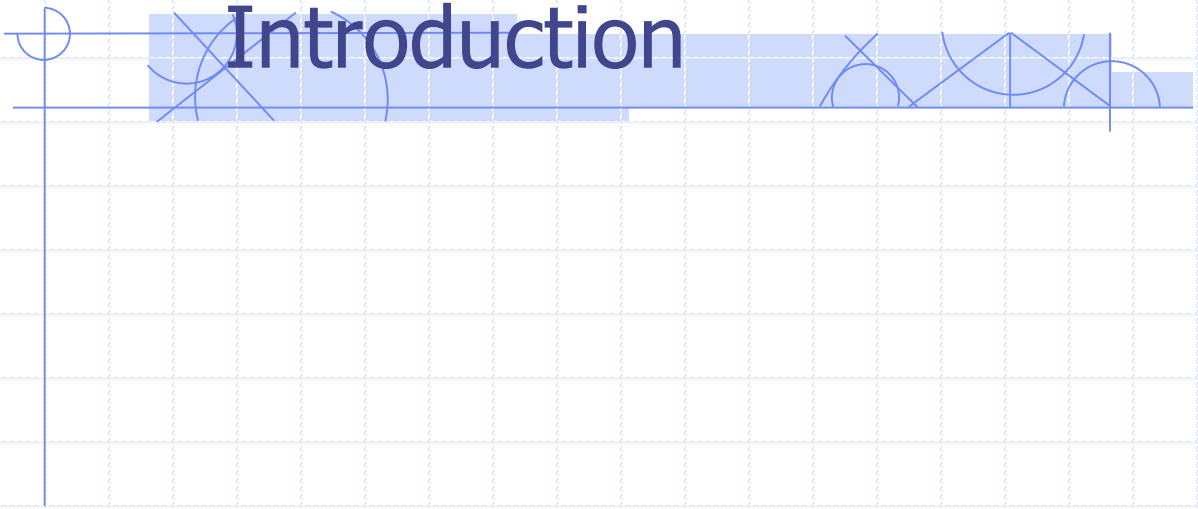
Graphical User Interface

Introduction



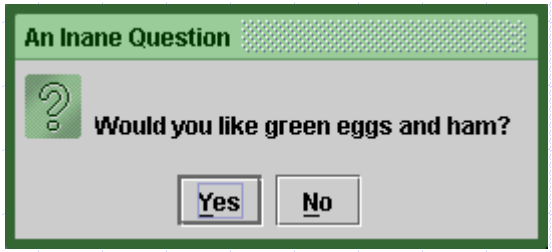
Graphical User Interface

- In Java, GUI-based programs are implemented by using classes from the **javax.swing** and **java.awt** packages.
- The Swing classes provide greater compatibility across different operating systems. They are fully implemented in Java, and behave the same on different operating systems.

Swing Components

- **Top-Level Containers**
  The components at the top of any Swing containment hierarchy.
  General-Purpose Containers
  Intermediate containers that can be used under many different circumstances.
- **Special-Purpose Containers**
  Intermediate containers that play specific roles in the UI.
- **Basic Controls**
  Atomic components that exist primarily to get input from the user; they generally also show simple state.
- **Uneditable Information Displays**
  Atomic components that exist solely to give the user information.
- **Interactive Displays of Highly Formatted Information**
  Atomic components that display highly formatted information that (if you choose) can be modified by the user.

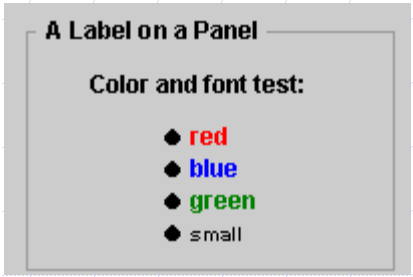
Top-Level Containers



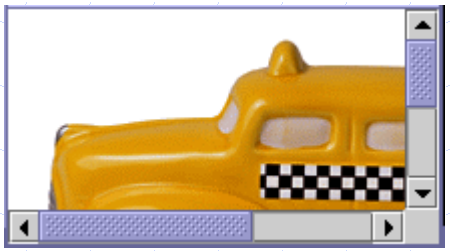
Dialog

Frame

General-Purpose Containers

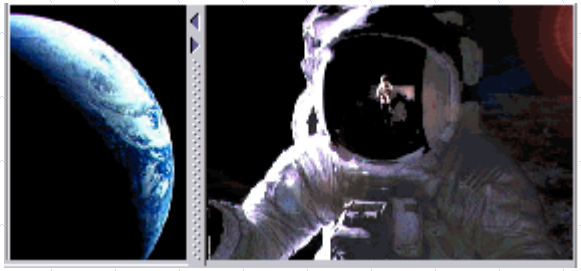


Panel



Scroll pane

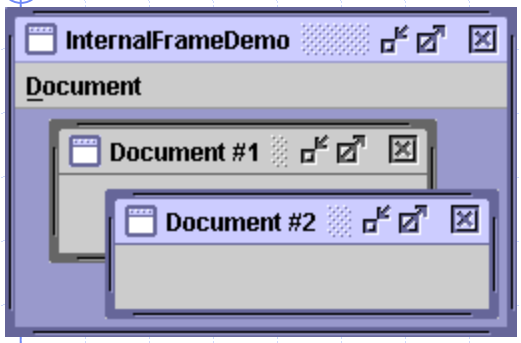
Tool bar



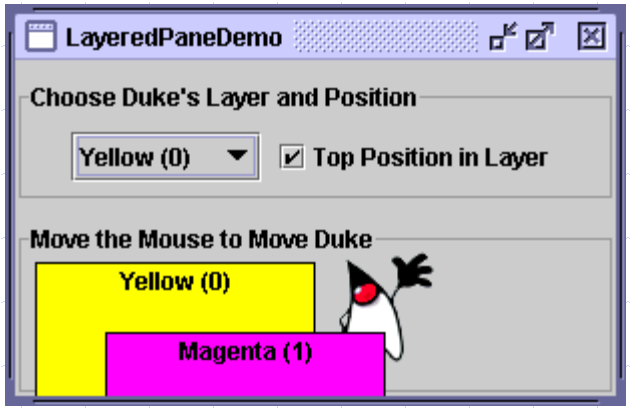
Split pane

Tabbed pane

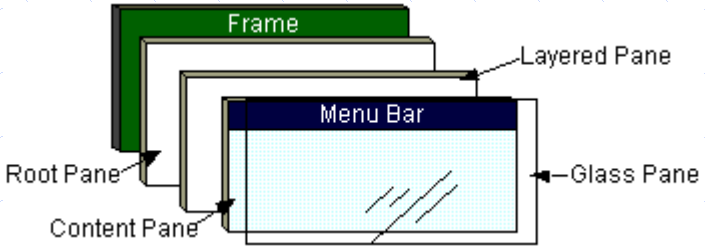
Special-Purpose Containers



Internal frame



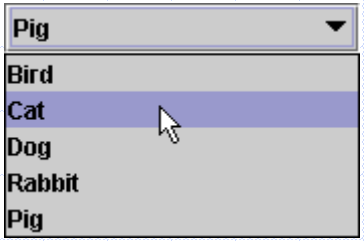
Layered pane



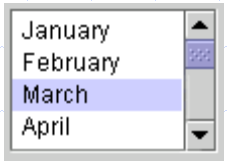
Root pane

Basic Controls

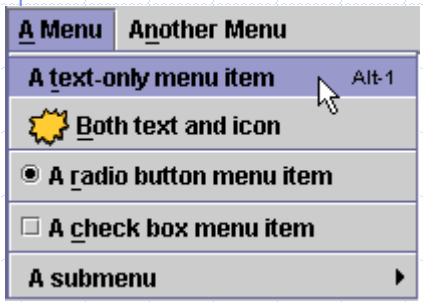
Buttons



Combo Box



List



Menu

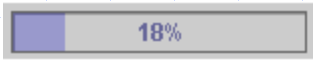
Slider

Text field
or
Formatted text field

Spinner

Uneditable Information Displays

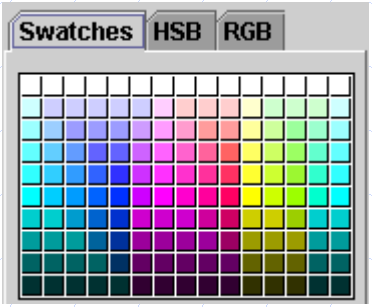
Label



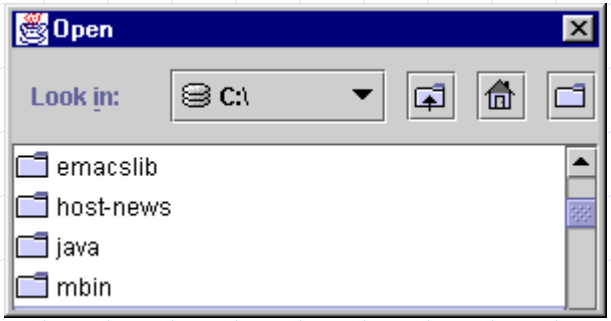
Progress bar

Tool tip

Interactive Displays of Highly Formatted Information



Color chooser



File chooser

First Name	Last Name	Favorite Food
Jeff	Dinkins	
Ewan	Dinkins	
Amy	Fowler	
Hania	Gajewska	
David	Geary	

Table

- red
- blue
- green
- small
- large
- *italic*
- **bold**

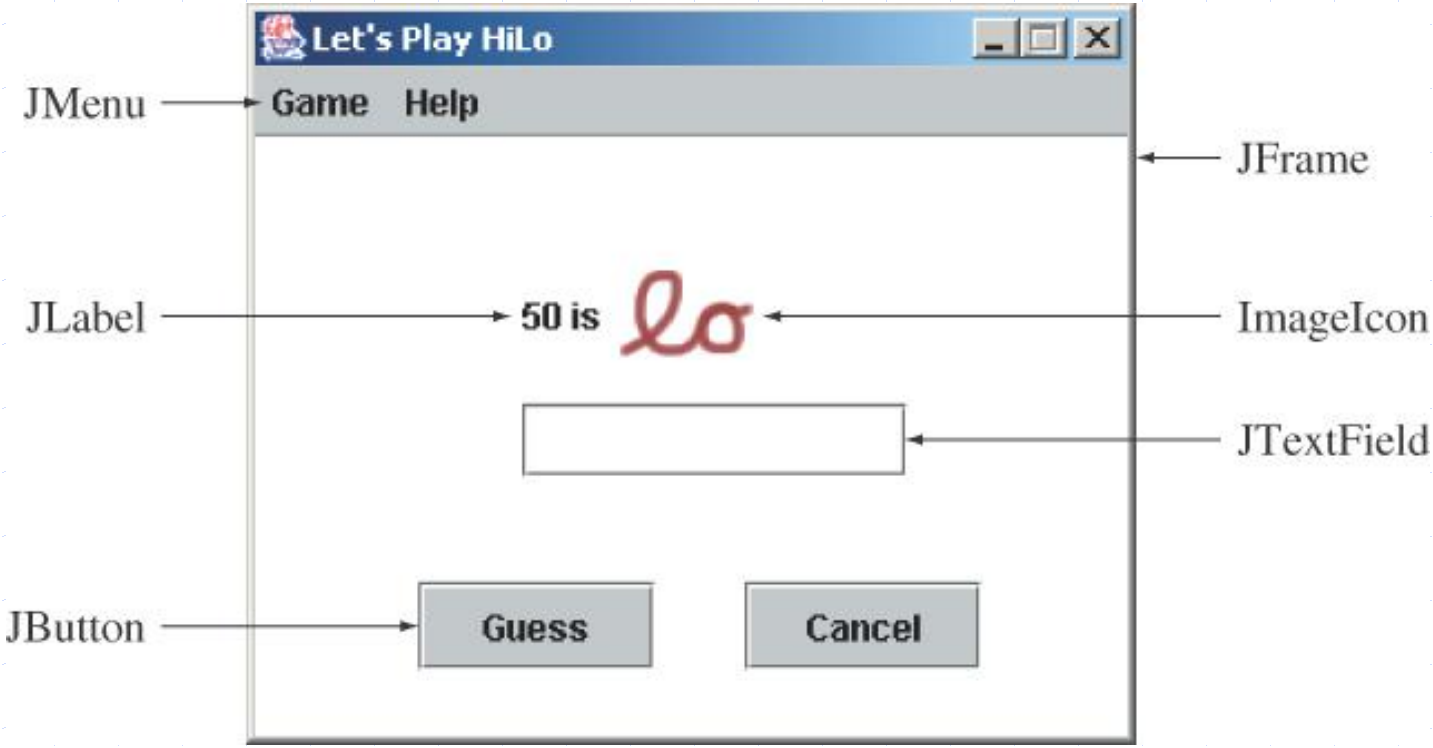
Text

- Music
  - Classical
    - Beethoven
    - Brahms
    - Mozart
  - Jazz
  - Rock

Tree

Sample GUI Objects

- Various GUI objects from the **javax.swing** package.

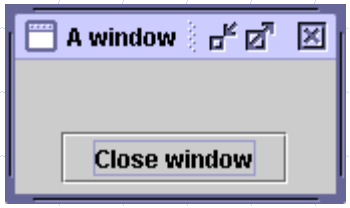


How to Make Frames (Main Windows)

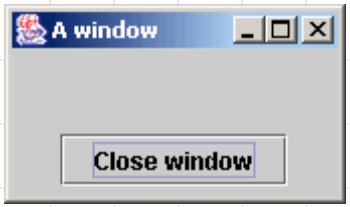
- //1. Optional: Specify who draws the window decorations.
  - **JFrame.setDefaultLookAndFeelDecorated(true);**
- //2. Create the frame.
  - **JFrame frame = new JFrame("FrameDemo");**
- //3. Optional: What happens when the frame closes?
  - **frame.setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE**);
- //4. Create components and put them in the frame. *...create emptyLabel...*
  - **frame.getContentPane**().**add(emptyLabel**, **BorderLayout.CENTER**);
- //5. Size the frame.
  - **frame.pack**();
- //6. Show it.
  - **frame.setVisible(true**);

Specifying Window Decorations

```
JFrame.setDefaultLookAndFeelDecorated(true);
```




```
JFrame.setDefaultLookAndFeelDecorated(false);
```



Creating a Plain **JFrame**

```
import javax.swing.*;
class SalamFrame {
    JFrame frame;
      public  void createAndShowGUI() {
    JFrame.setDefaultLookAndFeelDecorated(true);
    frame = new JFrame("Salam");
 frame.setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);
    JLabel label = new JLabel("Assalamo Alaikom");
        frame.getContentPane().add(label);
    frame.pack();
        frame.setVisible(true);
    }
}
```

Displaying the Window

```
import javax.swing.*;
class myMain {
     public  static void main(String[] rags) {
       SalamFrame myFrame = new SalamFrame();
       myFrame.createAndShowGUI();
    } //end main.
}
```

Subclassing **JFrame**

- To create a customized frame window, we define a subclass of the **JFrame** class.

- The **JFrame** class contains rudimentary functionalities to support features found in any frame window.

Creating a Subclass of **JFrame**

- To define a subclass of another class, we declare the subclass with the reserved word **extends**.

```
import javax.swing.*;

class myJFrameSubclass1 extends JFrame {
    . . .
}
```

Creating a Plain **JFrame**

```
import javax.swing.*;
class SalamJFrame extends JFrame {

    public  SalamJFrame() {
       super("Salam");
       JFrame.setDefaultLookAndFeelDecorated(true);
       setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);
       JLabel label = new JLabel("Assalamo Alaikom");
       getContentPane().add(label);
       pack();
    //  setVisible(true);
    }
}
```

Displaying the Window

```
import javax.swing.*;
class myMain {
     public  static void main(String[] rags) {
       SalamJFrame myFrame = new SalamJFrame();
       myFrame.setVisible(true);
    } //end main.
}
```

Customizing myJFrameSubclass1

- An instance of myJFrameSubclass1 will have the following default characteristics:
  - The title is set to **My First Subclass**.
  - The program terminates when the close box is clicked.
  - The size of the frame is 300 pixels wide by 200 pixels high.
  - The frame is positioned at screen coordinate (150, 250).
- These properties are set inside the default constructor.

Creating a Plain **JFrame**

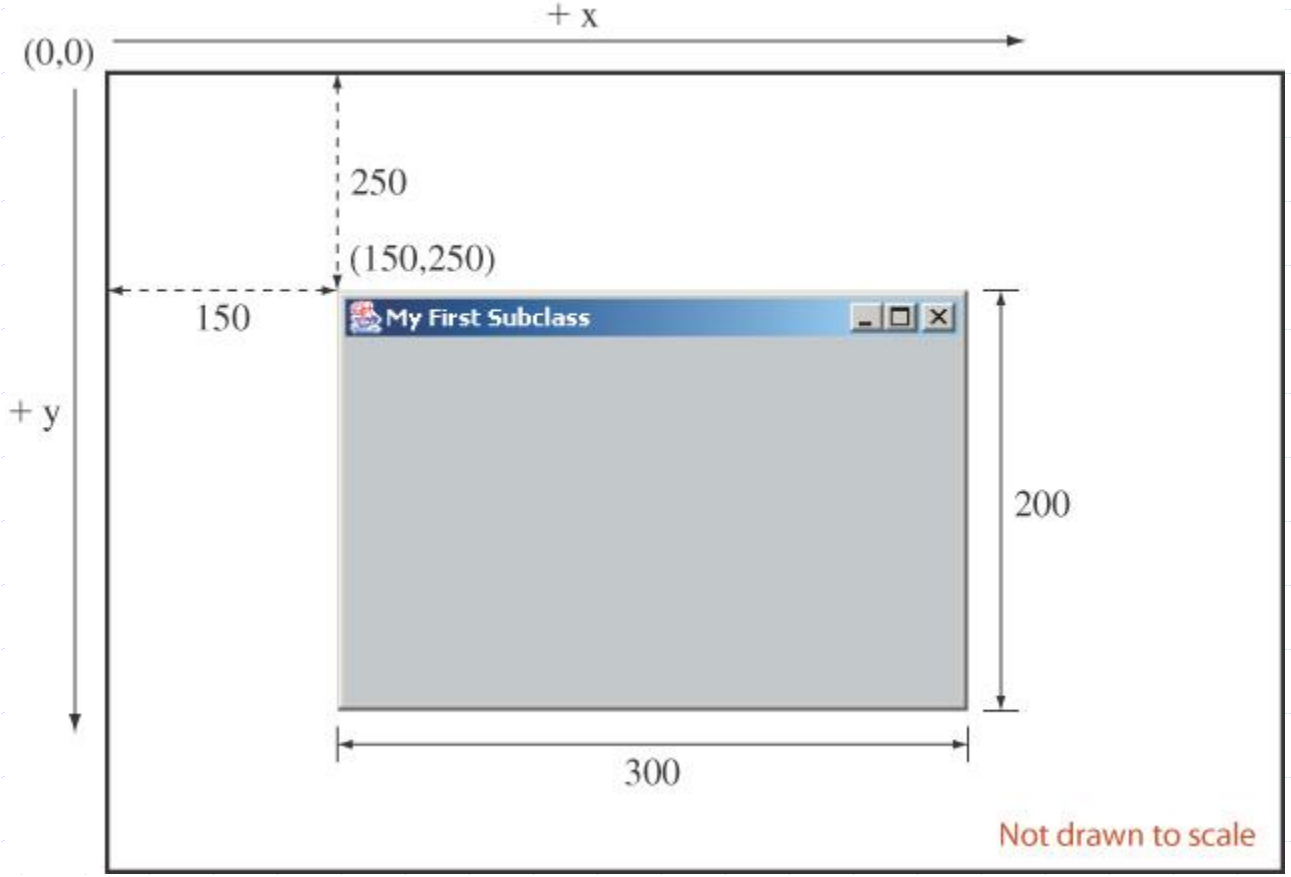
```
import javax.swing.*;
class SalamJFrame extends JFrame {

    public  SalamJFrame() {
       super("My First Subclass");
       JFrame.setDefaultLookAndFeelDecorated(true);
       setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);
       JLabel label = new JLabel("Assalamo Alaikom");
       getContentPane().add(label);
       setBounds(150, 250, 300, 200);
       pack();
    //  setVisible(true);
    }
}
```

Displaying the Window

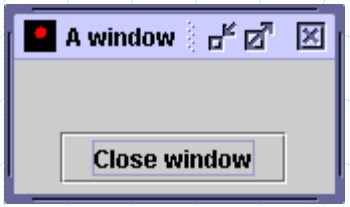
```
import javax.swing.*;
class myMain {
     public  static void main(String[] rags) {
    SalamJFrame myFrame = new SalamJFrame();
       myFrame.setVisible(true);
    } //end main.
}
```

Displaying myJFrameSubclass1

- Here's how a **myJFrameSubclass1** frame window will appear on the screen.



Creating and Setting Up a Frame

JFrame() JFrame(String)	Create a frame that is initially invisible. The String argument is the title for the frame.
void setDefaultCloseOperation(int) int getDefaultCloseOperation()	Set or get the operation that occurs when the user pushes the close button on this frame. Possible choices are: DO_NOTHING_ON_CLOSE HIDE_ON_CLOSE DISPOSE_ON_CLOSE EXIT_ON_CLOSE
void setIconImage(Image) Image getIconImage()	Set or get the icon that represents the frame. 

Creating and Setting Up a Frame (2)

void setTitle(String) String getTitle()	Set or get the frame's title.
void setUndecorated(boolean) boolean isUndecorated()	Set or get whether the window system should provide decorations for this frame. Works only if the frame is not yet displayable (hasn't been packed or shown).
static void setDefaultLookAndFeelDecorated(boolean) static boolean isDefaultLookAndFeelDecorated()	Determine whether subsequently created JFrames should have their Window decorations (such as borders, widgets for closing the window, title) provided by the current look and feel.

Setting the Window Size and Location

void pack()	Size the window so that all its contents are at or above their preferred sizes.
void setSize(int, int) void setSize(Dimension) Dimension getSize()	Set or get the total size of the window. The integer arguments to setSize specify the width and height, respectively.
void setBounds(int, int, int, int) void setBounds(Rectangle) Rectangle getBounds()	Set or get the size and position of the window. The window's upper left corner is at the x, y location specified by the first two args, and has the width and height specified by the last two args.
void setLocation(int, int) Point getLocation()	Set or get the location of the upper left corner of the window. The parameters are the x and y values, respectively.